AD-A058 107 TEXAS UNIV AT AUSTIN CENTER FOR CYBERNETIC STUDIES F/G 12/1

AN O(N(LOG N + LOG M)) ALGORITHM FOR LP KNAPSACKS WITH GUB CONS--ETC(U)

MAY 78 F GLOVER, D KLINGMAN N00014-75-C-0616

UNCLASSIFIED CCS-273 NL

1 OF 1

AD A058 107

END

DATE FILMED

10-78

DDC

LEVEL II

12

CENTER FOR CYBERNETIC STUDIES

The University of Texas
Austin, Texas 78712

14 CCS-273

9 Research Report, CCS 273

6 AN O(n(log n + log m)) ALGORITHM FOR LP KNAPSACKS WITH GUB CONSTRAINTS

by

10 Fred Glover*
Darwin Klingman**

11 May 1978 12 33 p.

* Professor of Management Science, University of Colorado, Boulder, CO 80302.

** Professor of Operations Research, Statistics, and Computer Sciences and Director of Computer Science Research, BEB 608, University of Texas at Austin, Austin, TX 78712

15 N00014-75-C-0569

This research was partly supported by ONR Contracts N00014-75-C-0616, N00014-75-C-0569, and N00014-78-C-0222 with the Center for Cybernetic Studies, The University of Texas. Reproduction in whole or in part is permitted for any purpose of the United States Government.

CENTER FOR CYBERNETIC STUDIES

A. Charnes, Director
Business-Economics Building, 203E
The University of Texas
Austin, Texas 78712
(512) 471-1821

ABSTRACT

A specialization of the dual simplex method is developed for solving the linear programming (LP) knapsack problem subject to generalized upper bound (GUB) constraints. The LP/GUB knapsack problem is of interest both for solving more general LP problems by the dual simplex method, and for applying surrogate constraint strategies to the solution of 0-1 "Multiple Choice" integer programming problems. We provide computational bounds for our method of $O(n(\log n + \log m))$, where n is the total number of problem variables and m is the number of GUB sets. In the commonly encountered situation where the number of variables in each GUB set is the same, our bound becomes $O(n \log n)$. These bounds reduce the previous best estimate of the order of complexity of the LP/GUB knapsack problem (due to Witzgall) and provide connections to computational bounds for the ordinary knapsack problem.

1. INTRODUCTION

A good deal of attention has been given to standard LP knapsacks for their role as relaxations in branch and bound methods for solving integer knapsack problems [2, 5, 9]. Such problems have been studied as an end in themselves, and also as surrogate constraint relaxations for more general 0-1 integer programming (IP) problems.

Many 0-1 IP problems, however, are of the "multiple choice" variety, attended by the requirement that the variables of partitioned subsets sum to one. Specialized IP methods for problems involving such generalized upper bound (GUB) constraints have been proposed in settings of varied generality (e.g., [3, 4, 6]), and recently some attention has been given to integer knapsacks with GUB constraints [14, 15]. To solve these and more general problems using LP and surrogate relaxations, it is important to be able to solve LP/GUB knapsacks efficiently. It is also valuable to be able to solve LP/GUB knapsack problems to accelerate the solution of ordinary LP/GUB problems by the dual simplex method, as pointed out by Witzgall [16]. Consequently, the goal of this paper is to develop an algorithm for the LP/GUB knapsack problem that is both easily implemented and highly efficient.

Two earlier papers dealing with this problem (in slightly less general form than treated here) are worthy of special note. The paper by Sinha and Zoltners [15] is the first to identify the characteristics of the undominated solution space for the case in which the knapsack is an inequality constraint. These authors then develop a method that is reported to speed the branch and bound solution of the integer GUB knapsack problem. The second paper, due to Witzgall [16], examines the case where the knapsack is an equality con-

straint spanned by the GUB sets. Witzgall's work is especially notable for its geometric characterizations and the specification of "worst case" computational bounds for his algorithm. In particular, the algorithm of [16] is shown to be of complexity $O(n \log n) + O(m(n-m))$, where n is the number of variables and m is the number of GUB sets. This is the first result that bounds the complexity of the LP/GUB knapsack problem in this manner.

In this paper we use an alternative framework that focuses directly on properties of the dual simplex method applied to the LP/GUB knapsack problem. After specifying necessary and sufficient conditions for dual feasible bases, we identify relationships that hold automatically in the application of the dual simplex method. These relationships are then utilized to develop a specialized version of this method which is shown to be of complexity at most $O(n(\log n + \log m))$, or in the case where each GUB set contains the same number of elements, $O(n \log n)$. These bounds are interesting not only because they reduce the previous estimate of the order complexity of the LP/GUB knapsack problem, but also because they reduce to the same form as one of the standard algorithmic bounds for the ordinary LP knapsack problem without GUB constraints, thereby establishing a connection between these more and less general problems.

2. PROBLEM NOTATION

The LP/GUB knapsack problem may be written

$$\text{Minimize} \quad \sum_{j \in N} c_j x_j \tag{1}$$

$$\text{subject to } \sum_{j \in N} a_j x_j = a_o \tag{2}$$

$$\sum_{j \in J_k} x_j = 1;\ k \in M = \{1,\ldots,m\} \tag{3}$$

$$x_j \geq 0;\ j \in N = \{1,\ldots,n\} \tag{4}$$

where $J_p \cap J_q = \emptyset$ for $p \neq q$ and $J = \bigcup_{k \in M} J_k \subset N.$

There are no restrictions on any of the problem coefficients (a_o, a_j, c_j), except that we exclude the trivial situation in which $a_j = 0$ for $j \in N - J$.

Two subcases of interest included by our results are for $N = J$ (as in Witzgall [16]) and for $N - J = \{n\}$, where x_n is a slack or surplus variable (as in Sinha and Zoltner [15]). We will comment on the specializations of our results to these subcases at appropriate points.

To begin, we make a simple and well known observation concerning the structure of basic solutions for this problem.

<u>Remark 1</u>. In every basic solution to the equations (2) and (3), $m - 1$ of the sets J_k, $k \in M$ will have exactly one basic variable. The remaining J_k set will have one basic variable if there is a basic variable in $N - J$, and otherwise will have two basic variables. (By convention we refer to a variable as "in" a set if its subscript is in the set.)

To facilitate the subsequent development, we will introduce notational conventions that will be useful for depicting the form of a typical basic solution within the framework of the dual simplex method. Throughout this paper, we will let J_q denote the exceptional set that has two basic variables,

when this situation applies, and in general, let x_{k*} denote the basic variable (or one of the basic variables) in set J_k, $k \in M$. We will suppose that k* is unique for each set J_k, and call x_{k*} the <u>starred basic variable</u> for J_k. In the case of J_q, we will denote the basic variable other than x_{q*} by $x_{q'}$. As will be seen, this convention will allow us to associate different formulas with x_{q*} and $x_{q'}$, though of course these formulas yield equivalent expressions when q* and q' are interchanged. Additionally when there exists a basic variable in N-J it is denoted by x_p. Finally, we introduce the objective function variable $x_o = -\sum_{j\in N} c_j x_j$ whose maximization achieves the minimization of (1), and let NB denote the index set of current nonbasic variables. (4)

<u>Basic solution forms</u>

<u>Case 1</u>. x_p is basic in N - J.

$$x_o + \sum_{j\in NB} u_j x_j = u_o \tag{5}$$

$$x_p + \sum_{j\in NB} v_j x_j = v_o \tag{6}$$

$$x_{k*} + \sum_{j\in NB\cap J_k} x_j = 1 \qquad k \in M \tag{7}$$

(Note, $NB \cap J_k = J_k - \{k^*\}$.)

<u>Case 2</u>. No variables are basic in N - J; $x_{q'}$ and x_{q*} are basic in J_q.

$$x_o + \sum_{j\in NB} u_j x_j = u_o \tag{8}$$

$$x_{q'} + \sum_{j\in NB} v_j x_j = v_o \tag{9}$$

$$x_{q*} + \sum_{j\in NB\cap J_q} (1-v_j)x_j + \sum_{j\in NB-J_q} (-v_j)\, x_j = 1 - v_o \tag{10}$$

$$x_{k*} + \sum_{j \varepsilon NB \cap J_k} x_j = 1 \qquad k \varepsilon M - \{q\} \tag{11}$$

For our subsequent development, we need to identify the precise connections between the coefficients of the basis representations in Case 1 and Case 2 and the coefficients of the original problem representation (1)-(4).

To reduce all formulas to the same notation for both Case 1 and Case 2 when $N \neq J$, we define $J_{m+1} = (N-J) \cup \{n+1\}$, where x_{n+1} is a "fictitious" variable, unrestricted in sign, with $a_{n+1} = c_{n+1} = 0$. We further specify that x_{n+1} is always the starred basic variable for the set J_{m+1}, i.e., $n+1 = (m+1)*$. Although we are completely unconcerned about the value of x_{n+1}, we may view x_{n+1} as definitionally equal to $1 - \sum_{j \varepsilon N-J} x_j$ and indeed x_{n+1} will receive this value by the prescriptions we will specify. Upon defining $M_o = M \cup \{m+1\}$ when $N \neq J$ and $M_o = M$ otherwise, the GUB equations of (3) therefore hold with M replaced by M_o. (That is, the existence of x_{n+1} would make Case 1 equivalent to Case 2 except for the fact that x_{n+1} is unrestricted.) In particular, then, the preceding equations for the Case 2 basic solution may be regarded as also applicable to Case 1, for $q = m + 1$, $q' = p$ and $q* = n + 1$, enabling subsequent formulas to be simplified. However, we will on occasion find it useful to discuss Case 1 and Case 2 on separate terms (when the unrestricted value of x_{n+1} has special implications).

By these conventions, the connections between the current basis coefficients and the original problem coefficients are expressed in the following remark.

Remark 2. Let $\alpha = a_o - \sum_{k\varepsilon M_o} a_{k*}$, $\delta = a_{q'} - a_{q*}$

and $d_j = c_j - v_j(c_{q'} - c_{q*})$

Then the coefficients of (8) - (11) (with M replaced by M_o) may be expressed in terms of those of (1) - (4) by:

$$v_o = \alpha/\delta$$

$$v_j = (a_j - a_{k*})/\delta \text{ for } j \ \varepsilon\ J_k,\ k \ \varepsilon\ M_o$$

$$u_j = d_j - c_{k*} \text{ for } j \ \varepsilon\ J_k,\ k \ \varepsilon\ M_o$$

The derivation of the remark is immediate by the application of Gaussian elimination. It may be noted, incidentally, that the arbitrary designation of $x_{q'}$ and x_{q*} implies that the coefficients of equation (10) can alternately be obtained from the expression for the v_j coefficients in Remark 2 by interchanging q' and q* in this expression.

3. PROPERTIES OF BASIC DUAL FEASIBLE SOLUTIONS

The goal of this section is to identify special properties of basic dual feasible solutions to (1) - (4), as a foundation for initialing a dual method. The following theorem (which slightly generalizes results of [15, 16]) accomplishes this by providing necessary and sufficient conditions for a basis to be dual feasible--i.e., to yield $u_j \geq 0$, $j \ \varepsilon$ NB, in the expression for x_o in (5) and (8). For this result we keep Case 1 and Case 2 separate.

Theorem 1. A basic solution is dual feasible for (1) - (4) if and only if:

k* is selected so that $d_{k*} = \underset{j\varepsilon J_k}{\text{Minimum}} \{d_j\} \qquad k \ \varepsilon\ M - \{w\}$,

(where d_j is as defined in Remark 2 and in Case 2, $\{w\} = \{q\}$ and $\{w\} = \emptyset$, otherwise).

Case 1. $c_h/a_h \leq c_i/a_i \qquad h \in H,\ i \in I$

where $H = \{h \in N - J: a_h < 0\}$, $I = \{i \in N - J: a_i > 0\}$ and p is selected to be an $h \in H$ that yields the maximum c_h/a_h or to be an $i \in I$ that yields the minimum c_i/a_i.

Case 2. q' is selected so that $c_{q'} \leq c_j$ for all $j \in J_q$ such that $a_j = a_{q'}$ and

$$\left.\begin{array}{ll} (c_{q'} - c_r)/(a_{q'} - a_r), & r \in R \\ \\ c_h/a_h, & h \in H \end{array}\right\} \leq \left\{\begin{array}{ll} (c_{q'} - c_s)/(a_{q'} - a_s), & s \in S \\ \\ c_i/a_i, & i \in I \end{array}\right.$$

where $R = \{r \in J_q : a_r < a_q'\}$, $S = \{s \in J_q : a_s > a_{q'}\}$.

Then q* is selected to be an $r \in R$ that yields the maximum value of all terms on the left of the foregoing inequality, or an $s \in S$ that yields the minimum of all terms on the right of the inequality, provided this is possible in consideration of the terms c_h/a_h and c_i/a_i. (Otherwise, the choice of q' does not allow dual feasibility. Also, whenever H or I is empty, the inequalities involving the corresponding $h \in H$ or $i \in I$ are not applicable.)

Proof. The stipulations about k* and Case 1 are immediate from Remark 2, noting that $c_{q*} = a_{q*} = 0$ for Case 1. The stipulations about Case 2 are derived as follows. When $x_{q'}$ and x_{q*} are both basic, then there are dual multipliers Θ for equation (2) and π for the J_q equation of (3) such that

$u_j = c_j - (\Theta a_j + \pi)$ for $j \in J_q$. These multipliers must be selected to yield $u_{q'} = u_{q*} = 0$. From $u_{q'} = 0$ we obtain $\pi = c_{q'} - \Theta a_{q'}$, and hence $u_j = c_j - c_{q'} + \Theta(a_{q'} - a_j)$. The dual feasibility requirement $u_j \geq 0$ yields

$$\Theta(a_{q'} - a_j) \geq c_{q'} - c_j$$

Thus, if $a_j = a_{q'}$, then $c_{q'} \leq c_j$, as first stipulated under Case 2. The alternatives $a_j < a_{q'}$ and $a_j > a_{q'}$, identified by $j \in R$ and $j \in S$, respectively, yield

$$(c_q - c_r)/(a_{q'} - a_r) \leq \Theta \leq (c_{q'} - c_s)/(a_{q'} - a_s) \quad r \in R,\ s \in S$$

Dual feasibility requirements $u_j = c_j - \Theta a_j \geq 0$ for $j \in N - J$ further yield

$$c_h/a_h \leq \Theta \leq c_i/a_i \qquad h \in H,\ i \in I$$

leading to the full set of inequalities stipulated for Case 2. Finally, the condition $u_{q*} = 0$ requires that $q*$ be selected so that $\Theta = (c_{q'} - c_{q*})/(a_{q'} - a_{q*})$. This completes the proof.

Theorem 1 discloses what may also be argued by simple dominance considerations--first, that we may throw out all elements of a set J_k with tied a_j values except for one with the smallest c_j value, and second, that all elements of H and I may be discarded except those yielding the maximum c_h/a_h and the minimum c_i/a_i. Thus $N - J$ can be restricted to at most two elements. If both these elements exist, and $c_h/a_h > c_i/a_i$, then the problem has an unbounded optimum. Otherwise, Case 1 of Theorem 1 provides an immediate starting dual feasible basic solution whenever $N - J$ is nonempty, by selecting either x_h or x_i as a basic variable (according to which of these variables exist). This observation also applies when $N = J$, because it is possible to

add an <u>artificial</u> variable x_n (for n increased by 1), yielding $N - J = \{n\}$, with $a_n = 1$ and c_n large. (This variable is not to be confused with the "fictitious" x_{n+1}.)

However, Theorem 1 also makes it possible to obtain starting dual feasible solutions without resorting to the elementary Case 1 situation. The following corollary indicates an easy way to do this when $N = J$ and $N - J = \{n\}$. We assume for this setting that $a_n = 1$ for $N - J = \{n\}$. In addition, we will suppose $c_n = 0$ for $N - J = \{n\}$, using Gaussian elimination on the objective function to achieve this if necessary.

<u>Corollary 1</u>. When $N = J$ or $N - J = \{n\}$, a Case 2 starting basic dual feasible solution can be obtained by designating any J_k to be J_q, selecting q' so that

$$a_{q'} = \underset{j \varepsilon J_q}{\text{Minimum}} \{a_j\}, \qquad c_{q'} = \underset{j \varepsilon J_q :\, a_j = a_q}{\text{Minimum}} \{c_j\}$$

and selecting $q^* \varepsilon S$ so that

$$(c_{q^*} - c_{q'})/(a_{q^*} - a_{q'}) = \underset{s \varepsilon S}{\text{Minimum}} (c_s - c_{q'})/(a_s - a_{q'})$$

If $S = \emptyset$, then $x_{q'} = 1$ (and the problem shrinks). If $S \neq \emptyset$, but $N - J = \{n\}$ (with $a_n = 1$ and $c_n = 0$), then $c_{q^*} < c_q$, or else, again $x_{q'} = 1$. (For this case $c_j \geq c_{q'}$ for $j \varepsilon J_q$ allows $x_j = 0$.)

When $N = J$ in Corollary 1, replacing (2) by its negative leads to an alternative application of the corollary, equivalent to picking $a_{q'}$ to be a maximum and selecting $q^* \varepsilon R$ to yield a maximum ratio.

We now turn to the main results of this paper, characterizing the relationships of the dual simplex method applied to (1) - (4), and developing

an efficient specialization for this problem. As a by-product we will also identify ways to generate other starting basic solutions that accord with the conditions of Theorem 1.

4. SPECIALIZATION OF THE DUAL SIMPLEX METHOD

For convenience in the following development, we outline the steps of the dual simplex method as follows.

Step 0. Begin with a dual feasible basis.

Step 1. Select any equation, other than the x_o equation, with a negative constant term. (If none exists, the current basic solution is optimal.) Represent this equation in the form of (9) (thereby identifying the outgoing variable as $x_{q'}$):

$$x_{q'} + \sum_{j \in NB} v_j x_j = v_o \quad (v_o < 0)$$

Step 2. Let $NB^- = \{j \in NB: v_j < 0\}$. If NB^- is empty, the problem has no feasible solution. Otherwise, select the incoming variable x_i, $i \in NB^-$ to yield

$$u_i/v_i = \underset{j \in NB^-}{\text{Maximum}} \{u_j/v_j\}$$

where the u_j coefficients are those of the current x_o equation (8).

Step 3. Execute a basis exchange (pivot) step that replaces $x_{q'}$ by x_i in the basis. The updated form of the pivot equation (9), which becomes the new x_i equation, is

$$x_i + \sum_{j \in NB^*} (v_j/v_i) x_j = v_o/v_i$$

where NB* is the new set of nonbasic variables (replacing i by q') and $v_{q'} = 1$ (as implicit in (9)). The updated form of all remaining equations is obtained by Gaussian elimination (or equivalently, direct substitution) using the x_i equation to remove x_i from the other equations. Then return to Step 1.

The foregoing description of the dual method is entirely general and not specific to the LP/GUB knapsack problem except for the notation linking the current pivot equation to (9) and the x_o equation to (8). By means of this notational link, however, we may now make additional observations concerning the solution path of the dual simplex method for this problem.

Note first of all that the convention of representing the pivot equation in the form of (9) is entirely permissible in the restricted setting of the LP/GUB knapsack problem since we may always interchange the roles of $x_{q'}$ and x_{q*} as necessary to allow this representation. Clearly, too, at most one of the two equations (9) and (10) can have a negative constant term and thereby qualify as the pivot equation. Thus, representing the pivot equation in the form of (9) serves to uniquely identify the indexes q' and q*. In fact, using the connections of Remark 2, we may immediately express the conditions for identifying $v_j < 0$ and the maximum ratio of Step 2 of the dual method in terms of the original problem coefficients.

Remark 3. If $a_{q'} > a_{q*}$, then

$$v_j < 0 \text{ if and only if } a_j < a_{k*} \tag{12}$$

and if in addition $v_j \neq 0$, $v_h \neq 0$ for $j \in J_r$, $h \in J_u$ (possibly r = u), then

$$u_j/v_j \leq u_h/v_h \text{ if and only if } \theta_{jr*} \leq \theta_{hu*} \tag{13}$$

where $\theta_{fg} = (c_f - c_g)/(a_f - a_g)$. If $a_{q'} < a_{q*}$, then the direction of the second

inequality in (12) and in (13) is reversed.

Although this remark follows directly by substituting the coefficient identities of Remark 2 into Remark 3, its implications are quite useful. This is due to the somewhat surprising fact that the application of the dual simplex method assures that if $a_{q'} > a_{q*}$ holds at one iteration, then $a_{q'} > a_{q*}$ (for other indexes q' and q*) at all iterations. This relationship and others associated with it are expressed in the following main result of this section.

<u>Theorem 2</u>. Let J_t denote the set containing the incoming variable x_i determined in Step 2 of the dual simplex method. If $t = m + 1$ (i.e., if $i \varepsilon N-J$), then the pivot must yield an optimal solution. If $t \leq m$, and if the pivot does not yield an optimal solution, then upon representing the next pivot equation also as (9), all of the following hold:

(a) J_t becomes the new J_q

(b) x_{t*} becomes the new outgoing variable $x_{q'}$

(c) x_i becomes the new x_{q*}

(d) the ratio values Θ_{jk*}, $j \varepsilon J_k$, remain unchanged for all $k \varepsilon M_o - \{t\}$

(e) $a_{q'} > a_{q*}$ before the pivot if and only if $a_{q'} > a_{q*}$ (for the new q' and q*) after the pivot.

(f) Over a series of pivots, as the index k is periodically selected as t, the elements a_{k*} will only change in descending sequence if $a_{q'} > a_{q*}$ and will only change in ascending sequence if $a_{q'} < a_{q*}$.

<u>Proof</u>. Each of the assertions is a direct outcome of applying the dual simplex method. First, the x_i equation of Step 3 of the dual method must have a posi-

tive constant term (since both v_o and v_i are negative), and cannot qualify as the new pivot equation. However, this equation currently has the form of (9) (since x_{t*} and not x_i, is the current starred basic variable for the set J_t). Thus, equation (10) is the only possibility for the new pivot equation, in which case it may be put in the form of (9) by interchanging the roles of i and t*. The interchange of i and t* is unnecessary if $i \in N-J$ because x_{t*} is the unrestricted variable x_{n+1}, and an optimal solution is already obtained. Otherwise, if the current solution is not feasible (the solution value of x_i exceeds 1), the interchange immediately establishes (a), (b) and (c) of the theorem. Next, since J_t is the only set J_k in which the identity of x_{k*} changes by the pivot, it also follows that the values Θ_{jk*} change only for k = t, establishing (d). The condition $a_{q'} > a_{q*}$ before the pivot is equivalent to stipulating $a_i < a_{t*}$ in consideration of the fact that $v_i < 0$ (Remark 3). But since t* becomes the new q' and i becomes the new q*, this yields (e). Finally, (f) follows directly from (e) and Remark 3, completing the proof.

We will henceforth suppose for simplicity that $a_{q'} > a_{q*}$ on all iterations, understanding that the directions of inequalities specified in the following discussion may have to be reversed if this is not the case. (Alternatively, it is always possible to assure $a_{q'} > a_{q*}$ by the device of replacing equation (2) by its negative in case $a_{q'} < a_{q*}$.) With this understanding, Theorem 2 directly implies

<u>Corollary 2</u>. (For $a_{q'} > a_{q*}$): If the maximum ratio R_k, given by

$$R_k = \underset{\substack{j \varepsilon J_k \\ a_j < a_{k*}}}{\text{Maximum}} \{\Theta_{jk*}\} \qquad (14)$$

is known for each set J_k, $k \varepsilon M_o$, together with the index $i(k)$ such that $R_k = \Theta_{jk*}$ for $j = i(k)$, then the incoming variable x_i is identified by

$$i = i(t) \text{ where } R_t = \underset{k \varepsilon M_o}{\text{Maximum}} \{R_k\} \qquad (15)$$

and the pivot step leaves all R_k except R_t unchanged for the determination of the new R_t by (15) at the next pivot. (If $a_{q'} < a_{q*}$, the maximum in (14) is replaced by a minimum over $a_j > a_{k*}$.)

The significance of Corollary 2 is twofold. First of all, it allows the dual simplex method to be implemented for the LP/GUB knapsack problem without ever explicitly calculating the u_j and v_j coefficients. Secondly, it allows the R_k values to be efficiently stored in a heap, with the maximum R_t at the top. Then as R_t is removed, and replaced with a new value, the unchanged values of the remaining R_k enable the heap to be reconstituted with minimal computation (on the order of $O(\log m)$).

The issue remaining before giving a detailed specification of the steps of a specialized dual algorithm, is the efficient determination of R_k by (14). Since each time a new R_k is found, the variable $x_{i(k)}$ will become the new x_{k*} (the next time k is selected as t by (15)), all of the $j \varepsilon J_k$ such that $a_j \geq a_{i(k)}$ may immediately be dropped, since they will be of no further interest. This approach by itself, as will be shown, leads to a specialized method whose worst case computational bound is superior to that of [16] when the number of GUB sets exceeds the number of elements in each set. (This generally occurs

in practical applications of an "assignment" nature, where the number of items to be assigned generally far exceeds the number of possible assignments per item.) However, an even better approach from the standpoint of worst case bounds results by a simple preliminary pass through each set J_k, eliminating in advance the elements that do not qualify to be selected as k*. Since the elements that are left will be visited in descending order of the the a_j values (for $a_{q'} > a_{q*}$), it follows that each sucessively smaller a_j will be the next a_{k*}, and the task of identifying a maximum by (14) is eliminated.

Specifically, then, we seek to identify a subset J_k^o of J_k whose elements are linked by a predecessor/successor ordering, where the immediate successor s(j) of an index $j \in J_k^o$ identifies the next element that qualifies to serve as k* after j, and the immediate predecessor p(j) of j identifies the element of J_k^o that qualifies to serve as k* immediately before j. Initially, of course, s(j) and p(j) just arrange the elements of J_k in descending (ascending) order and we will suppose that in the process of creating such a linking that duplicate a_j values are removed by retaining only the one associated with the smallest c_j value. The process of dropping an element from J_k in the construction of J_k^o can be accomplished simply by linking its immediate predecessor to its immediate successor.

Under this predecessor /successor linking, (14) can be written

$$R_k = \Theta_{ik*} \geq \Theta_{jk*} \text{ for all successors } j \text{ of } i = i(k)$$

Then i will become the new k* (except for the first i selected as k*). Thus, in particular, since we may eliminate the situation of tied maximum ratios by selecting the one with the smallest a_i coefficient (which has no tied successors),

and since dropping superfluous elements will yield $k^* = p(i)$, the identifying characteristic of J_k^o becomes

$$\Theta_{ip(i)} > \Theta_{jp(i)} \tag{16}$$

for all successors j of i and for all $i \in \overline{J}_k^o$, where $\overline{J}_k^o$ is J_k^o stripped of its first and last elements, which respectively have no predecessors or successors. The task of weeding out elements of J_k to assure this relationship is made easy by the following.

Remark 4. The inequality (16) holds for all $i \in \overline{J}_k^o$ and for all successors j of i if and only if it holds for all $i \in \overline{J}_k^o$ and for $j = s(i)$.

Proof. We need only show that for any h, i, j, r (taking the roles $h = p(i)$, $j = s(i)$ and $r = s(j)$) such that $a_h > a_i > a_j > a_r$, the two "successive" inequalities $\Theta_{ih} > \Theta_{jh}$ and $\Theta_{ji} > \Theta_{rj}$ imply $\Theta_{ih} > \Theta_{rh}$. First, for the coefficients as ordered, we note that $\Theta_{ih} > \Theta_{jh}$ is equivalent to $\Theta_{jh} > \Theta_{ji}$, since both of these inequalities reduce to $c_h a_i + c_i a_j + c_j a_h > c_i a_h + c_j a_i + c_h a_j$. Similarly, $\Theta_{ji} > \Theta_{rj}$ is equivalent to $\Theta_{rj} > \Theta_{ri}$. Hence we obtain $\Theta_{ih} > \Theta_{jh} > \Theta_{ji} > \Theta_{rj} > \Theta_{ri}$ and in particular $\Theta_{jh} > \Theta_{rj}$, which is equivalent to $\Theta_{rj} > \Theta_{rh}$. Consequently, $\Theta_{ih} > \Theta_{rh}$, completing the proof.

To make convenient use of this observation we introduce a dummy index 0 to "start" and "terminate" the predecessor/successor linking, where 0 is treated as the immediate predecessor of the largest a_j and the immediate successor of the smallest a_j. The procedure for modifying the initial linking on J_k so that it becomes a linking on J_k^o is then as follows.

0. To start, let h, i and j be the "first three" elements of J_k, that is,

$h = s(0)$, $i = s(h)$, $j = s(i)$. (If J_k has less than three elements, then $J_k^o = J_k$ and nothing is to be done.)

1. Compare Θ_{ih} to Θ_{jh}.

   (a) If $\Theta_{ih} > \Theta_{jh}$, set $h = i$ and go to Step 2.

   (b) If $\Theta_{ij} = \Theta_{jh}$ or if $\Theta_{ih} < \Theta_{jh}$ and $p(h) = 0$, drop i and go to to Step 2.

   (c) If $\Theta_{ih} < \Theta_{jh}$ and $p(h) \neq 0$, drop i, set $i = h$, and $h = p(i)$. Then return to the start of Step 1.

2. Set $i = j$ and $j = s(i)$. If $j = 0$, the procedure stops and the linking correctly identifies the ordered elements of J_k^o. Otherwise, return to Step 1.

The validity of the foregoing procedure is an immediate consequence of Remark 4. Note that the index j never "backs up" to a predecessor value, but remains unchanged in Step 1 and is set to its successor at Step 2. Consequently Step 2 will always be executed $n_k - 2$ times, where n_k is the number of elements in J_k. Whenever the method does not go to Step 2, the index i is dropped at 1(c), which can occur at most $n_k - 2$ times (since i is never the first or last element), for a total number of iterations of the procedure equalling at most $2(n_k - 2)$. This procedure is patterned after one due to Witzgall [16] (who obtains a different iteration count) except that Witzgall's approach is based upon a geometric determination of the locations of points on or below line segments, rather than on a direct comparison of ratios as afforded by Remark 4.

It should also be noted, in contrast to the less general situation examined in [16], that the elements of J_k^o may not all qualify to be basic in a

dual feasible solution. If $N \neq J$, it is additionally necessary that the ratios $\Theta_{ip(i)}$ be bounded by the limiting ratios from $N - J$, as shown in Theorem 1. This means that some of the initial and final elements of J_k^o (under the predecessor/successor linking) may also drop out of consideration. Rather than bothering to check for this situation in advance, however, the first and last relevant elements of J_k^o can be determined automatically by starting from some initial basic dual feasible solution and simply executing the specialized dual algorithm.

In general, these observations lead to the following Corollary as an extension of the options available from Corollary 1 for obtaining an initial dual feasible basis.

Corollary 3. The set of Case 2 dual feasible bases, any one of which provides an acceptable starting basis for the specialized dual simplex method, can be generated by selecting an arbitrary J_k^o to be J_q^o, and selecting any element i from this set (other than the first element) such that $\Theta_{ip(i)}$ satisfies the limiting bounds from $N - J$ (identified in Theorem 1). Then i and p(i) may respectively serve as q' and q*. If no such element i exists, then some other set must serve as J_q^o , and whatever element of the "unacceptable" J_k^o thereby enters the basis in the starting solution is compelled to be basic in all dual feasible bases (hence, the associated variable may be fixed at the value 1).

The elements q' and q* found in Corollary 3 may need to be interchanged, so that the first pivot equation can be represented by (9). (In this case, the $a_{q'} > a_{q*}$ assumption must be replaced by the $a_{q*} > a_{q'}$ assumption, reversing the roles of the predecessor/successor links.) If a Case 1 basis is used as the start, then $a_p > 0$ (for x_p the basic variable in (5)) implies

$a_{q'} > a_{q*}$ on all iterations (since p takes the initial role of q' with $a_{q*} = a_{n+1} = 0$), whereas an artificial start (with $p = n$, $a_n = 1$ and c_n large) will select the first ("largest a_j") element of each J_k as the initial a_{k*}.

The specialized dual simplex method based on the foregoing results may now be described as follows.

The Specialized Dual Simplex Method

1. **Initialization**.

   (a) Create the predecessor/successor linkings and the J_k^o sets, $k \in M_o$. (For $N \neq J$, define J_{m+1}^o to be the set containing the elements (at most two in number) with limiting ratios identified by Theorem 1.) (This step can be deferred or applied in conjunction with Step 1(b), using the starting basis there to reduce the range of elements considered for inclusion in the J_k^o sets.)

   (b) Create a starting dual feasible basis (as by Theorem 1 and Corollary 1 or Corollary 3). Compute the initial v_o value by computing

   $$\alpha = a_o - \sum_{k \in M} a_{k*} \text{ and } v_o = \alpha/(a_{q'} - a_{q*})$$

   If $v_o \geq 0$ and either $q' \in N - J$ or $v_o \leq 1$, then the current basic solution ($x_{q'} = v_o$, $x_{q*} = 1 - v_o$ and $x_{k*} = 1, k \in M_o - \{q\}$) is optimal. Otherwise, interchange q' and q* if necessary so that $v_o < 0$. For what follows we suppose $a_{q'} > a_{q*}$. (If not, the word "maximum" should be replaced by "minimum," and the successor symbol s() should be replaced by the predecessor symbol p().)

(c) Identify the ratios $R_k = \Theta_{s(k^*)k^*}$ for each $k \in M_o$. (If $s(k^*) = 0$, the ratio R_k does not exist, and is bypassed. For the case $k = m + 1$ where by convention $k^* = n + 1$, we define $s(k^*)$ to be the first element of J^o_{m+1} excluding the current q' (if $q' \in J_{m+1}$). Hence $R_{m+1} = c_j/a_j$ for $j = s(k^*)$, if this element j exists.) Put these ratios in a heap, with the maximum at the top.

2. Identify the incoming basic variable and the new basis composition.

Pick the maximum ratio from the top of the heap and denote it R_t. (If the heap is empty, there is no feasible solution.) The current variable $x_{q'}$ leaves the basis and $x_{s(t^*)}$ enters the basis. If $s(t^*) \in N - J$, the current basic solution is optimal for $q' = s(t^*)$ and $v_o = \alpha/(-a_{q'})$ (where α is unchanged from its previous value). Otherwise, the current x_{t^*} becomes the new $x_{q'}$, while $x_{s(t^*)}$ is the new x_{q^*}; i.e., set $q' = t^*$ and $q^* = t^* = s(t^*)$.

3. Update the current basic solution.

Update α and v_o by setting $\delta = a_{q'} - a_{q^*}$, $\alpha = \alpha + \delta$ and $v_o = \alpha/\delta$. If $v_o \geq 0$, the current basic solution is optimal. Otherwise, identify the new value of $R_t = \Theta_{s(t^*)t^*}$ (for the new t^*). If the ratio does not exist ($s(t^*) = 0$), reform the heap for the ratios still in it. Otherwise, add R_t back to the heap. Then return to Step 2.

An analysis of the maximum amount of computation required by this method is an follows. The creation of the predecessor/successor linkings (that initially arrange the a_j coefficients in descending/ascending order for each J_k) requires on the order of $O(n_k \log n_k)$ computation for each set, or an effort

of at most $\sum_{k \in M_o} O(n_k \log n_k) \leq O(n \log n)$. (For the case where each GUB set has the same number of elements, n/m, we may refine this to $O(n(\log n - \log m))$.)

The work to modify the linking to identify the J_k^o set involves at most $2n_k - 4$ iterations of the procedure based on Remark 4, or $2n - 4m$ iterations over all sets, requiring computation or order $O(n - m) \leq O(n)$. Creating the starting feasible basis requires an effort of at most $O(n)$ (including the effort of generating and selecting the minimum d_j values) while computing the initial v_o value is $O(m)$. Finally, computing the R_k ratios requires $O(m)$ computation, while putting them in a heap is an effort of order $O(m \log m)$. Thus, the total initialization effort of Step 1 can be expressed as $O(n \log n) + O(m \log m)$.

For Steps 2 and 3, at most $n - m - 1$ elements (the successors of the k* elements, excluding the initial q') remain to be examined in the J_k^o sets, $k \in M_o$ and so these steps will require at most $n - m - 1$ iterations. Exclusive of reforming the heap, these two steps require a handful of "if checks," assignments, a couple of additions and 1 division. Reforming the heap requires an effort of $O(\log m)$, hence in total the amount of effort required at Steps 2 and 3 is $O((n - m)\log m)$. Putting these together with the effort required at initialization we can state

<u>Theorem 3</u>. The computational complexity of the LP/GUB knapsack problem is of order at most

$$O(n \log n) + O(n \log m)$$

or

$$O(n(\log n + \log m))$$

We have stated these order bounds separately instead of simply giving the $O(n(\log n + \log m))$ bound, because of the overestimate involved in the $O(n \log n)$ term. In particular, as previously noted, this term can instead be expressed as $O(n(\log n - \log m))$ for the situation in which each GUB set has n/m elements. Thus, in this case we have

Corollary 4. When each GUB set contains the same number of elements, the computational complexity of the LP/GUB knapsack problem is at most $O(n \log n)$.

The bound of Witzgall is given in [16] as $O(n \log n) + O((n - m)m)$, where the $O(n \log n)$ term is essentially the same as that of Theorem 3, and also can be replaced by $O(n(\log n - \log m))$ for GUB sets with n/m elements. The primary difference between the bound of [16] and that of Theorem 3 is therefore the contrast between $O((n - m)m)$ and $O(n \log m)$. For easier comparison, let $g = n/m$ (so that g is the number of elements in each GUB set if each set has the same cardinality). Then these terms can be respectively written $O((g - 1)m^2)$ and $O(g(m \log m))$. Since $g \geq 2$ in any meaningful problem (or else there are GUB sets with only 1 element), the latter term clearly represents a smaller order of effort than the former, particularly as m or g (hence n) becomes larger. This difference appears to stem from the fact that our procedure specializes the dual simplex method directly, whereas Witzgall's instead carries out preliminary "topological reductions" (corresponding to those obtained via Remark 4) but otherwise leaves the dual method primarily to its own devices (for the case $J = N$). (Sinha and Zoltner's procedure and Witzgall's procedure appear closely related in this respect.)

It is interesting to note the type of order bound that results for our method when the initialization effort of setting up the predecesor/successor

links and adapting them to the J_k^o sets is not employed.

The modifications for this approach are as follows.

Alternative method. (Omitting the initial ordering of the a_j coefficients by the predecessor/successor links, and the creation of the J_k^o sets.)

1. Initialization.

   (a) Deleted

   (b) As in the previous method, except that Corollary 3 is not used as a strategy for creating an initial basis. In addition, drop the index k^* from each J_k.

   (c) Instead of setting $R_k = \Theta_{s(k^*)k^*}$, examine each $j \in J_k$ (for J_k as currently constituted). If $a_j \gtrsim a_{k^*}$ then drop j from J_k, and if not compute the ratio Θ_{jk^*}, saving the minimum of these computed ratios as R_k. (Then $s(k^*)$ denotes the j that gives this minimum ratio.)

2. Identify the incoming basis variable and the new basis composition.

   As in the previous method.

3. Update the current basic solution.

   As in the previous method, except for setting $R_t = \Theta_{s(t^*)t^*}$. Instead, first drop t^* from J_t, and for each remaining $j \in J_t$ (as currently constituted) carry out the operation indicated in 1(c) for $k = t$.

The type of analyses applied to the computations for the previous method allows us to state

<u>Corollary 5</u>. When each GUB set has the same number of elements $g = m/n$, the computational effort required by the Alternative Method for the LP/GUB knapsack problem is of order

$$O(n) + O((n - m)\log m) + O((n - m)g)$$

or

$$O(n) + O((n - m)(g + \log m)$$

Again we have written the bound in different ways to facilitate comparison with the other bounds. The $O(n)$ term here is comparable to a $O(n)$ term that was previously assimilated into $O(n \log n)$ in both our approach and in Witzgall's. Thus, for a clearer comparison, the bound of Corollary 4 can be rewritten

$$O(n) + O(n \log n)$$

and that of Witzgall can be written

$$O(n) + O(n \log g) + O((n - m)m).$$

While the worst case bound of Corollary 4 appears generally superior to the other two, note that the bound for the Alternative Method appears more attractive than that of [16] for $g = m$, and becomes increasingly attractive as m becomes larger relative to g, due to the fact that increases in $\log m$ are dwarfed by increases in m. (The value of m is often several fold greater than g in practical applications. For example, in the applications of [8, 11, 12, 13], m ranges from $4g$ to $50g$.) Coupling this with the fact that the Alternative Method requires less "set up" effort than the other methods makes

it an appealing alternative for problems in which worst case bounds are expected to be overly pessimistic. In this context, any attempt to consider "likely" cases instead of worst cases must also account for the advantages that may derive from initiating a specialized dual algorithm from an advanced starting basis, rather from an "extreme end" of the dual feasible region (as in [15] and [16]).

Finally, it is interesting to consider the specialization of these bounds to the ordinary knapsack problem. In this problem, the number of variables before adding slacks to give GUB constraints is $m = n/2$ (i.e., the addition of slacks yields $g = 2$). Bounds of <u>both</u> previously indicated versions of the Specialized Dual Simplex Method (from Corollary 4 and Corollary 5) reduce to $O(m \log m)$ in this case, which is a standard bound for algorithms for the knapsack problem. Recently, however, Balas and Zemel [1] have developed an improved bound of $O(m)$ for m variable knapsacks. This raises the interesting question of whether it is possible to find a method for the general LP/GUB knapsack problem whose worst case computational effort specializes to $O(m)$, yet that maintains advantages for the general case. We conjecture that this is not possible.

ACKNOWLEDGEMENT

The authors are indebted to Christoph Witzgall for conversations on this problem and for sending us a copy of his recent paper [16], which motivated our identification of computational bounds.

REFERENCES

1. Balas, E. and Zemel, E. "Solving Large 0-1 Knapsack Problems," *Management Sciences Report No. 408*, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, June 1976, Revised July 1977.

2. Barr, Richard S. and Ross, G. T. "A Linked List Data Structure for a Binary Knapsack Algorithm," Research Report CCS 232, Center for Cybernetic Studies, The University of Texas at Austin, August 1975.

3. Beale, E. M. L. and Tomlin, J. A., "Special Facilities in a General Mathematical Programming System for Non-Convex Problems Using Special Ordered Sets of Variables," in *Proceedings of the 5th IFORS Conference*, London, 1970, 447-454.

4. Beale, E. M. L. and Tomlin, J. A., "An Integer Programming Approach to a Class of Combinatorial Problems," *Mathematical Programming*, 3, 3, (1972), 339-344.

5. Fayard, D. and Plateau, G., "Resolution of the 0-1 Knapsack Problem: Comparison of Methods," *Mathematical Programming*, (1975), 272-307.

6. Glover, F., "Convexity Cuts for Multiple Choice Problems," *Discrete Mathematics*, 6 (1973), 221-234.

7. Glover, F., "Surrogate Constraint Duality in Mathematical Programming," *Operations Research*, 23 (1975), 434-451.

8. Glover, F. and Klingman, D., "Network Applications in Industry and Government," *AIIE Transactions*, December 1977.

9. Greenberg, H. and Hegerick, R. L., "A Branch Search Algorithm for the Knapsack Problem," *Management Science*, 16 (1970), 327-332.

10. Greenberg, H. and Pierskalla, W. P., "Surrogate Mathematical Programming," *Operations Research*, 18 (1970), 924-939.

11. Gross, D., and Pinkus, C. E., "Optimal Allocation of Ships to Yards for Regular Overhauls," *Technical Report 63095*, George Washington University, Washington, D.C., 1972.

12. Klingman, D., Ross, G. T., and Soland, R. M., "Optimal Lot-Sizing and Machine Loading for Multiple Products," Proceeding of Conference on the Disaggregation Problem, Ohio State University, Columbus, Ohio, March 1977.

13. Masud, Enver , "An Interactive Procedure for Sizing and Timing Distribution Substations Using Optimization Techniques," *Paper T75 142-6*, Presented at the IEEE PES Winter Meeting, New York, January 27-February 1, 1974.

14. Nauss, Robert M., "The 0-1 Knapsack Problem with Multiple Choice Constraints," Working Paper, University of Missouri, St. Louis, March 1975, revised May 1976.

15. Sinha, P. and Zoltners, A. A., "The Multiple Choice Knapsack Problem," *Research Report No. 8-76*, School of Business Administration, University of Massachusetts, September 1976.

16. Witzgall, Christoph, "On One-Row Linear Programs," Applied Mathematics Division National Bureau of Standards, Washington, D.C., May 1978.

Unclassified
Security Classification

DOCUMENT CONTROL DATA - R & D		
(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)		
1. ORIGINATING ACTIVITY (Corporate author) Center for Cybernetic Studies The University of Texas	2a. REPORT SECURITY CLASSIFICATION Unclassified	
	2b. GROUP	
3. REPORT TITLE AN 0(n(log n + log m)) Algorithm for LP Knapsacks with GUB Constraints		
4. DESCRIPTIVE NOTES (Type of report and inclusive dates)		
5. AUTHOR(S) (First name, middle initial, last name) Fred Glover Darwin Klingman		
6. REPORT DATE May 1978	7a. TOTAL NO. OF PAGES 30	7b. NO. OF REFS 16
8a. CONTRACT OR GRANT NO. CNR N00014-75-C-0616; 0569 ONR N00014-78-C-0222 b. PROJECT NO. NR047-021 c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) Center for Cybernetic Studies Research Report CCS 273	
	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	
10. DISTRIBUTION STATEMENT This document has been approved for public release and sale; its distribution is unlimited.		
11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Cffice of Naval Research (Code 434) Washington, D.C.	

13. ABSTRACT

A specialization of the dual simplex method is developed for solving the linear programming (LP) knapsack problem subject to generalized upper bound (GUB) constraints. The LP/GUB knapsack problem is of interest both for solving more general LP problems by the dual simplex method, and for applying surrogate constraint strategies to the solution of 0-1 "Multiple Choice" integer programming problems. ~~We provide~~ computational bounds ~~for our~~ method of 0(n(log n + log m)), where n is the total number of problem variables and m is the number of GUB sets. In the commonly encountered situation where the number of variables in each GUB set is the same, our bound becomes 0(n log n). These bounds reduce the previous best estimate of the order of complexity of the LP/GUB knapsack problem (~~due to Witzgall~~) and provide connections to computational bounds for the ordinary knapsack problem.

+ are provided for this

DD FORM 1473 1 NOV 65 (PAGE 1)
S/N 0101-807-6811

Unclassified
Security Classification

A-31408

Unclassified

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Knapsack						
GUB						
Multiple choice						
Surrogate constraints						
Dual method						